OIPE JC191
AUG 24 2005
PATENT & TRADEMARK OFFICE

ABSTRACT OF Korean Patent Application No. 10-1998-0060513

The present invention disclosed herein is an interface apparatus of encoding video data and a method thereof. The interface apparatus according to the present invention comprises a SRSM where data is inputted/outputted according to an address assignment; an address generating means generating an address for reading and writing the data of the SRAM; an input interfacing means storing data inputted from an encoding device to output it to the SDRAM; an output interfacing means for outputting the data stored in he SDRAM to an external channel; and a control signal generating means for controlling input/output of the data of the SDRAM. In accordance with the present invention, instead of a conventional asynchronous DRAM, a synchronous DRAM is adopted. As a result, the bit width and the complexity in an interface apparatus are reduced, thereby lowering cost. Accordingly, the cost of a video interface chip is lowered, and power consumption is also reduced.

AL

(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(51) Int. Cl.		(45) 공고일자	2001년05월02일
H04N 7/24		(11) 등록번호	10-0288142
		(24) 등록일자	2001년02월05일
(21) 출원번호	10-1998-0060153	(65) 공개번호	특2000-0043736
(22) 출원일자	1998년12월29일	(43) 공개일자	2000년07월15일

(73) 특허권자 한국전기통신공사 이계철
경기 성남시 분당구 정자동 206
(72) 발명자 김건수
대전광역시 서구 내동 220번지 롯데아파트 117-101
고종석
(74) 대리인 대전광역시 유성구 송강동 청솔아파트 205-1502
특허법인 신성 박해천, 특허법인 신성 원석희, 특허법인 신성 최중석, 특허법인 신성 박정후, 특허법인 신성 정지원

심사관 : 나경표

(54) 부호화 영상 데이터의 인터페이스 장치 및 그 방법

요약

1. 청구 범위에 기재된 발명이 속한 기술분야

본 발명은 비디오 인코더의 인터페이스 장치 및 그 방법에 관한 것임.

2. 발명이 해결하려고 하는 기술적 과제

본 발명은 비디오 인코더의 부호화된 비트스트림을 채널로 출력할 때 버퍼링을 위한 입출력 메모리(SRAM)의 크기를 줄여 하드웨어 비용을 절약하고, 소비 전력을 줄인 비디오 인코더의 인터페이스 장치 및 그 방법을 제공하고자 함.

3. 발명의 해결방법의 요지

본 발명에서는 기존의 디램(DRAM) 대신에 처리속도가 빠르고 제어가 간편한 에스디램(SDRAM)을 사용하고, 에스디램의 메모리맵을 효율적으로 구성하며, 메모리 어드레스와 제어 신호를 생성하는 장치를 별도로 두어 메모리 엑세스를 위한 하드웨어를 줄이며, 입출력 데이터를 읽고 쓸 때 제어신호의 생성을 통합하여 제어함으로써 하드웨어 구조를 단순화하였다.

4. 발명의 중요한 용도

본 발명은 비디오 인코더의 부호화된 비트스트림을 채널로 출력하기 위한 인터페이스에 이용됨.

대표도

도1

명세서

도면의 간단한 설명

도 1은 본 발명에서 제시하는 비디오 인코더의 부호화된 비트스트림 인터페이스 장치의 구성도.

도 2는 도 1의 비트스트림 인터페이스 장치 중 입력 인터페이스 장치의 블록도.

도 3은 도 2의 내부 구조도.

도 4는 도 3에 있는 SDRAM의 어드레스 포인터 구조도.

도 5는 입력 인터페이스 장치의 동작 타이밍도.

도 6은 도 1의 비트스트림 인터페이스 장치 중 출력 인터페이스 장치의 블록도.

도 7은 도 6의 내부 구조도.

도 8은 도 7에 있는 SDRAM의 어드레스 포인터 구조도.

도 9은 출력 인터페이스 장치의 동작 타이밍도.

도 10은 도 1의 비트스트림 인터페이스 장치 중 제어신호 생성기의 블록도.

도 11은 도 10의 내부 구조도.

도 12는 제어신호 생성기의 동작 타이밍도.

도 13은 SDRAM 어드레스 포인터 구조도.

도 14는 도 1의 비트스트림 인터페이스 장치 중 SDRAM 어드레스 생성기의 블록도.

도 15는 도 14의 내부 구조도.

도 16은 SDRAM 어드레스 생성기의 동작 타이밍도.

도 17은 도 1의 비트스트림 인터페이스 장치 중 버퍼상태 연산기의 블록도.

도 18은 도 19의 내부 구조도.

도 19는 버퍼상태 연산기의 동작 타이밍도.

* 도면의 주요 부분에 대한 부호의 설명

100: 입력 인터페이스장치,	200: 출력 인터페이스장치,
300: 제어신호 생성기,	400: 어드레스 생성기,
500: 버퍼상태 연산기,	600: 에스디램,
700: 입력버퍼,	800: 출력버퍼,
900: 멀티플렉서.	

발명의 상세한 설명

발명의 목적

발명이 속하는 기술분야 및 그 분야의 종래기술

본 발명은 비디오인코더(Video Encoder)의 부호화(encoding)된 비트스트림(Bit Stream)을 채널(channel)로 출력하기 위한 인터페이스(interface)장치 및 그 방법에 관한 것으로, 특히 싱크로너스 다이나믹램(Synchronous Dynamic RAM; 이하 "에스디램(SDRAM)"이라 함)을 사용하는 인터페이스장치 및 그 방법에 관한 것이다.

MPEG-2 비디오 인코더의 하드웨어는 입력영상 처리모듈(IP module), 부호화기 제어모듈(control module), 움직임/추정보상모듈(MB&C module), 프레임 메모리모듈, DCT/양자화기 모듈 및 가변길이 부호화기 모듈(VLC module) 등과 같이 6가지 모듈로 나눌 수 있다.

이 중에서 프레임 메모리모듈의 인터페이스는 입력영상 처리모듈로부터의 데이터 저장 및 출력, 부호화된 영상의 일시적인 저장, 움직임 추정/보상을 위해 필요한 영상(프레임/필드)의 저장과 출력, 비트율 조정을 위한 부호화된 비트열의 저장 및 출력 등의 역할을 한다. 프레임 메모리 설계는 이들 각 모듈들과의 인터페이스에 필요한 입출력 버퍼의 크기를 가능한 적게 사용하여 각 모듈들이 데이터를 처리할 수 있도록 타이밍을 맞추는 것이 주요점이 된다.

여기서, 종래에는 프레임 메모리모듈을 비동기식(Asynchronous) 다이나믹램을 여러개 사용하여 설계하였다. 이러한 구조에서는 메모리의 데이터폭이 64비트이며, 인터페이스 비트폭이 커 배선의 복잡도가 심하여 레이아웃(lay-out) 면적이 크게 되는 문제점이 있었다. 또한, 비디오 인코더 내부의 데이터버퍼링을 위한 에스램(SRAM)의 소요가 많아서 결과적으로 하드웨어 구현상 비용이 많이 드는 부담이 있었다.

발명이 이루고자 하는 기술적 과제

따라서, 본 발명은 상기 문제점을 해결하기 위해 안출된 것으로서, 하드웨어 구현 비용을 적게 하는 부호화 영상데이터의 인터페이스 장치 및 그 방법을 제공하는데 그 목적이 있다.

또한, 본 발명의 다른 목적은, 비디오 인코더의 부호화된 비트스트림을 채널로 출력할 때 버퍼링을 위한 입출력 메모리 및 제어장치를 간단화한 부호화 영상데이터의 인터페이스 장치 및 그 방법을 제공하는데 있다.

또한, 본 발명의 또다른 목적은, 인터페이스 배선의 복잡도를 간소화하면서 비디오 인코더 내부의 데이터 버퍼링을 위한 내부메모리의 실제 면적을 최소화한 부호화 영상데이터의 인터페이스 장치 및 그 방법을 제공하는데 있다.

발명의 구성 및 작용

상기 목적들을 달성하기 위한 본 발명의 일실시예는, 부호화 영상데이터의 인터페이스장치에 있어서, 어드레스 지정에 따라 데이터의 입/출력이 이루어지는 에스디램; 상기 에스디램의 데이터 리드 및 라이트를 위한 어드레스를 발생하는 어드레스 생성 수단; 부호화 장치로부터 입력된 데이터를 저장하여 상기 에스디램으로 출력하는 입력 인터페이싱 수단; 상기 에스디램에 저장된 데이터를 외부채널로 출력하는 출력 인터페이싱 수단; 및 상기 에스디램의 데이터 입력과 출력을 제어하는 제어신호 생성 수단을 구비하는 것을 특징으로 한다.

상기 다른 목적을 달성하기 위한 본 발명의 다른 실시예는, 부호화 영상데이터의 인터페이스장치에 있어

서, 어드레스 지정에 따라 데이터의 입/출력이 이루어지는 에스디램; 상기 에스디램의 데이터 리드 및 라이트를 위한 어드레스를 발생하는 어드레스 생성 수단; 부호화 장치로부터 입력된 데이터를 저장하여 상기 에스디램으로 출력하는 입력 인터페싱 수단; 상기 에스디램에 저장된 데이터를 외부채널로 출력하는 출력 인터페이싱 수단; 상기 에스디램의 데이터 입력과 출력을 제어하는 제어신호 생성 수단; 및 현재의 채널버퍼에 저장된 데이터의 정보를 이용하여 가변길이 부호화의 속도를 결정하는 버퍼상태 연산 수단을 구비하는 것을 특징으로 한다.

또한, 상기 본 발명의 목적들을 달성하기 위하여 본 발명은, 부호화 영상데이터의 인터페이스 방법에 있어서, 에스디램을 이용하여 어드레스 지정에 따른 데이터의 입/출력을 수행하는 제 1 단계; 상기 에스디램의 데이터 리드 및 라이트를 위한 어드레스를 발생하는 제 2 단계; 부호화 장치로부터 입력된 데이터를 저장하여 상기 에스디램으로 출력하는 제 3 단계; 상기 에스디램에 저장된 데이터를 외부채널로 출력하는 제 4 단계; 상기 에스디램의 데이터 입력과 출력을 제어하는 제어신호를 생성하는 제 5 단계; 및 현재의 채널버퍼에 저장된 데이터의 정보를 이용하여 가변길이 부호화의 속도를 결정하는 제 6 단계를 포함하는 것을 특징으로 한다.

상기의 특징적 실시구성에 의하면, 본 발명에서는 기존의 디램(DRAM) 대신에 처리속도가 빠르고 제어가 간편한 에스디램(SDRAM)을 사용하고, 에스디램의 메모리맵을 효율적으로 구성하며, 메모리 어드레스와 제어신호를 생성하는 장치를 별도로 두어 메모리 액세스를 위한 하드웨어를 줄이며, 입출력 데이터를 읽고 쓸 때 제어신호의 생성을 통합하여 제어함으로써 하드웨어 구조를 단순화 하였다. 그리고, 버퍼 상태를 계산하는 장치를 효율적으로 구성하고, 에스디램의 인터페이스 데이터 폭을 32 비트로 묶여 하드웨어 라우팅의 복잡도를 줄였다. 즉, 본 발명은 비디오 인코더의 부호화된 비트스트림을 채널로 출력할 때 버퍼링을 위한 입출력 메모리(SDRAM) 및 제어 장치의 단순화를 통해서 하드웨어 비용을 절약할 수 있으며, 이에 대하여는 상세히 후술될 것이다.

이하, 본 발명이 속하는 기술분야에서 통상의 지식을 가진 자가 본 발명의 기술적 사상을 용이하게 실시할 수 있을 정도로 상세히 설명하기 위하여, 본 발명의 가장 바람직한 실시예를 첨부된 도면을 참조하여 설명하기로 한다. 도면에서 종래기술과 동일한 구성요소에 대하여는 동일한 도면 부호를 인용하였다.

도 1은 본 발명에 의한 부호화된 비트스트림의 인터페이스 장치에 대한 전체 블록도이다. 도1의 구성은, 어드레스(address)지정에 따라 데이터의 입/출력이 이루어지는 메모리로서의 에스디램(600)과, 상기 에스디램(600)의 데이터 리드(read) 및 라이트(write)를 위한 어드레스를 발생하는 어드레스생성기(400)와, 부호화장치로부터 입력된 데이터를 저장하여 상기 에스디램(600)으로 출력하는 입력인터페이스장치(100)와, 상기 에스디램(600)에 저장된 데이터를 외부채널로 출력하는 출력인터페이스장치(200)와, 에스디램(600)의 데이터 입력과 출력을 제어하는 제어신호생성기(300)와, 현재의 에스디램에 저장된 데이터의 정보를 이용하여 가변길이 부호화의 속도를 결정하는 버퍼상태 연산기(500)가 주요 구성을 이루고 있다.

도1의 각 부의 특징을 살피면 다음과 같다.

도1에서 입력 인터페이스장치(100)의 기능은 크게 다음과 같이 두 가지로 나눌 수 있다. 먼저 첫 번째 기능은 매 매크로블록마다 부호화 장치로부터 입력된 가변 길의 비트스트림 데이터(vstrm(31:0))를 일시적으로 입력 버퍼(700)(이는 "VSRAM32×192"로 구현됨)에 저장하는 과정을 제어하는 것이다. 이렇게 하기 위해서 매 클럭(clock)마다 입력되는 데이터의 개수를 계산하여, 데이터 개수에 따른 입력버퍼(700)의 어드레스를 데이터와 함께 입력 버퍼(700)로 출력시켜서 입력된 데이터를 일시적으로 버퍼에 저장해야 한다. 두 번째 기능은 이렇게 입력된 데이터를 채널 버퍼(channel buffer)로 할당된 에스디램(SDRAM)(600)의 영역으로 출력하여 저장하는 것이다. 입력 버퍼(700)의 데이터 읽기 어드레스는 제어신호생성기(300)가 에스디램(600)을 사용하도록 허가해 주는 인에이블 신호가 '1'일 때마다 하나씩 증가한다. 입력 버퍼(700)에 저장된 데이터는 인가된 읽기 어드레스에 따라 에스디램(600)으로 하나씩 출력된다. 입력 버퍼(700)의 입출력 데이터 속도(rate)가 다른 경우에 (입력은 27㎒, 출력은 54㎒), 입력 버퍼(700)의 언더플로우(underflow : 데이터 엠프티)를 방지하기 위한 장치가 추가되었다.

그리고, 출력 인터페이스장치(200)의 기능은 입력 인터페이스장치(100)의 기능과 비슷하다. 다만 데이터의 흐름 방향이 반대이므로 외부로 향등률의 데이터(cstrm(7:0))를 출력시켜 주는 역할을 한다. 출력 인터페이스장치(200)의 기능은 크게 출력버퍼(800)(이는 "CSRAM32×64"로 실시구성됨)의 어드레스를 생성하고, 에스디램(600)에서 읽어낸 데이터의 개수만큼 일시적으로 출력 버퍼(800)에 데이터를 저장하는 부분과 버퍼에 저장되어 있는 데이터를 외부 채널로 출력시켜 주는 역할을 하는 두 부분으로 나누어 볼 수 있다. 이러한 두 가지 기능을 수행하기 위해서 입력 인터페이스장치(100)와 마찬가지로 제어신호생성기(300)로부터 인에이블신호를 입력 받아서 입력 데이터의 어드레스를 생성하고 데이터 입력을 제어한다. 출력 버퍼(800)에서의 데이터 출력은 외부 인에이블 신호에 따라 이루어진다. 또한, 출력 버퍼(800)의 오버플로우(overflow)를 막기 위한 장치가 함께 포함된다.

도1에서 에스디램(600)의 어드레스 생성기(400)는 메모리에 데이터를 쓰거나 읽어낼 때의 어드레스를 생성시켜 주는 장치이다. 어드레스의 생성은 제어신호 생성기(300)로부터 입력되는 인에이블 신호에 따라서 이루어진다. 생성된 어드레스는 에스디램(600)을 제어하는 장치로 출력되어 사용된다. 같은 구조를 가진 두 개의 장치 즉, 읽기어드레스생성기(400A)와 쓰기어드레스생성기(400B)를 사용하여 각각 데이터의 입력과 출력 어드레스를 따로 생성하도록 하였다. 비트스트림 인터페이스 장치는 생성된 입출력 어드레스를 부호화 장치의 스케쥴링 타이밍에 따라서 멀티플렉서(900)를 통해 멀티플렉싱(multiplexing)하여 제어신호 생성기(300)를 통하여 에스디램(600)으로 출력해준다.

한편, 제어신호 생성기(300)는 에스디램(600)의 제어신호(read/write)를 생성하는 역할을 한다. 도1에서 에스디램(600)의 제어신호는 RAS, CAS, WE, CS, DQM 등이 있다. 이러한 신호에 따라서 각각 로우활성화(row_activate), 로우 프리차지 또는 로우 비활성화(row_deactivate), 비동작(nop), 리드(read), 라이트(write) 등의 동작을 만들어 준다. 이러한 제어신호 외에도 에스디램(600) 어드레스 생성기로부터 입력받은 어드레스를 동시에 에스디램(600)으로 출력하여 데이터를 정해진 어드레스에 쓰거나 특정 어드레스로부터 데이터를 읽는 기능을 가지고 있다. 전체 장치의 구성은 내부 카운터(internal counter)에 의해 결정되는 유한상태기(FSM : Finite State Machine)를 따라 각 상태에서 정해진 제어 신호를 생성하도록 설

계되어 있다. 각각의 타이밍(timing)은 부호화 장치로부터 입력되는 인에이블 신호에 의해 결정된다.

버퍼상태 연산기(500)는 현재 입출력 버퍼(700,800)와 채널 버퍼(600)에 저장되어 있는 데이터의 개수를 바이트(byte) 단위로 계산하여 출력해 주는 모듈이다. 이렇게 계산된 버퍼의 상태는 부호화 장치로 출력되어 데이터의 부호화 속도(rate)의 결정에 사용된다. 버퍼의 상태는 매 매크로블록(macro block)마다 계산된다. 계산된 결과는 현재 버퍼의 상태를 출력하라는 인에이블 신호가 입력 될 때마다 외부로 출력된다.

이하, 본 발명에 의한 부호화 영상데이터의 인터페이스장치의 각 구성요소별로 상세히 그 실시구성을 설명하겠다.

(1) 입력 인터페이스 장치

입력 인터페이스장치(100)의 기능은 앞에서 설명한 바와 바와 같이 부호화 장치로부터 입력된 32 비트 데이터를 일시적으로 입력 버퍼(700)에 저장했다가 메모리(600)로 출력시켜 주는 것이다. 데이터의 입력은 인에이블 신호에 의존한다. 데이터를 출력할 때는 시퀀스 디코더로부터 메모리 액세스 허가를 얻어야 한다. 데이터가 입력되면 그에 대응하는 에스램 쓰기(SRAM write) 어드레스를 생성시켜주고 데이터를 해당하는 에스램(SRAM) 어드레스에 차례대로 저장한다. 메모리에 데이터를 저장해도 된다는 인에이블 신호가 입력되면 에스램(SRAM)의 읽기(read) 어드레스에 따라서 해당 데이터를 32 비트씩 출력시켜 준다. 입력되는 데이터는 27㎒에 동기되고 출력되는 데이터는 54㎒에 동기되는 경우에 데이터의 언더플로우(underflow)를 방지하는 기능이 추가되어 있다.

도 2는 전체 입력 인터페이스 장치의 입출력 구성을 나타낸 것으로, 주요 핀을 설명하면 다음과 같다.

vstrm_en : 유효한 데이터가 부호화 장치로부터 입력된다는 인에이블 신호.

WE_SDRAM : 메모리로 데이터를 출력하라는 인에이블 신호.

in_proh : 입력 데이터와 출력 데이터의 어드레스를 비교하여 데이터 overwrite를 방지하는 신호.

pre_zero_detect : underflow를 막기 위한 falg 신호.

여기서, 데이터는 vstrm_en 신호가 '1'인 동안에 27㎒에 동기되어 입력된다. vstrm_en 신호를 반 클럭 지연시킨 신호와 sck2(27㎒) 클럭을 이용하여 입력된 데이터의 어드레스를 계산한다. 어드레스는 입력 버퍼(700)의 크기인 0_191의 값을 가지는 8 비트 카운터를 이용하면 쉽게 구할 수 있다. 메모리로 출력되는 데이터는 제어신호 생성기로부터 인에이블 신호를 받아서 동작한다. 즉, WE_SDRAM 신호가 '0'일 때 증가하는 8 비트 카운터를 사용하면 계산할 수 있다. 에스램(SRAM)의 WE 신호는 입력되는 vstrm_en 신호와 sck2클럭을 이용하여 만들어 낸다. 실제 데이터는 SRAMWE 신호가 '1'에서 '0'으로 떨어지는 순간에 에스램(SRAM)으로 입력된다. 따라서, 이 때의 에스램(SRAM) 어드레스가 안정적이 되도록 설계하여야 한다.

도3은 도2의 내부 구조를 나타낸 것이다. 에스램(SRAM)의 데이터를 읽어낼 때는 별도로 OE 신호를 주지 않는다. 읽어야 할 데이터의 어드레스를 주면 약간의 지연 후에 데이터가 에스디램(SDRAM)으로 출력되는 것을 알 수 있다. 또한, 언더플로우(underflow)를 막기 위해서 사용하는 플래그(falg) 신호는 입력되는 데이터와 출력되는 데이터의 어드레스를 비교함으로써 생성할 수 있다. 여기서, 사용한 방법은 두 어드레스의 차이를 계산하여 그 차이값이 '1'이 되는 순간에 플래그(falg) 신호를 '1'로 세팅시켜 주는 방법이다. 이 신호는 제어신호 생성기로 입력되어 WE_SDRAM 신호를 '1'로 만들어 주고, 다음 클럭에서는 더 이상 에스램(SRAM)의 데이터를 액세스하지 않게 해준다. in_proh 신호는 에스램(SRAM) 입출력 데이터의 어드레스를 비교하여 같은 경우에 '1'을 제어신호 생성기로 출력시켜 준다. 이 신호는 다음 차례에 에스디램(SDRAM)에 액세스하는 타이밍이 주어질 때, 메모리 제어 신호의 출력을 제어하여 에스디램(SDRAM)의 오동작을 방지한다. 즉, 잘못된 데이터가 에스디램(SDRAM)에 써지는 것을 방지한다.

도4는 입력 버퍼(700)의 어드레스 포인터(address pointer) 구조이며, 이상의 과정을 나타내는 타이밍도를 도5에 나타내었다. 입력 데이터의 어드레스는 입력 데이터의 인에이블 신호인 vstrm_en 신호를 반클럭만큼 지연시킨 신호가 '1'일 때마다 증가한다. 입력 버퍼의 어드레스 크기 만큼의 카운터를 이용하여 구현하였고, 출력 데이터의 어드레스도 비슷한 방식으로 구현하였다. 여기서, 인에이블 신호인 delayed_we_sdram은 제어신호 생성기에서 생성시켜 주는 에스디램(SDRAM) 데이터 쓰기 인에이블(write enable) 신호를 반클럭 만큼 지연시켜 준 것이다. 입력 어드레스는 데이터가 입력될 때마다 입력되는 데이터의 개수를 셀 수 있게 해준다. 이렇게 입력된 데이터는 타이밍을 맞추어서 정확한 값이 입력 버퍼에 들어가도록 쓰기 인에이블(write enable) 신호를 생성시켜 줘야 한다. 먼저, sram_we_make라는 신호는 vstrm_en 신호가 '1'일 때마다 그 값이 반전된다. 그리고, 이 신호하고 이 신호가 반클럭만큼 지연시킨 신호를 delayed_sram_we_make라고 하면, 위에서 보이는 바와 같이 결합(combinational) 로직을 이용하여 sram_we 신호를 만들 수 있다. 이렇게 생성된 인에이블 신호를 에스램(SRAM)에 입력으로 주면, 어드레스를 증가시키면서 입력되는 데이터의 값이 차례로 입력 버퍼에 들어가게 된다. 저장되어 있는 데이터를은 읽기 어드레스(read_addr)가 주어지면 약간의 지연 후에 에스디램(600)으로 출력된다. 데이터의 출력은 어드레스를 인가하면 자동으로 되기 때문에 별도의 출력 인에이블(output enable) 신호를 만들어 줄 필요가 없다. 이와는 별도로 데이터 입출력의 오동작을 막기 위한 프로세스가 필요하다. in_proh 신호는 입력 버퍼에 들어오는 데이터의 수가 적을 경우를 대비한 것이다. 한 매크로블록 마다 에스디램(SDRAM)으로 출력되도록 정해진 데이터의 개수는 에스디램(SDRAM) 타이밍 스케쥴링에 의해 정해져 있는데, 입력되는 데이터가 적은 경우 에스램(SRAM)의 입출력 어드레스가 겹칠 수가 있다. 이 경우에 잘못된 데이터가 전해지지 않도록 방지해 주는 신호가 in_proh 이다. 이 신호가 '1'이 되면 제어신호 생성기가 in_proh 신호를 받아서 에스디램(SDRAM)의 제어신호를 변경하게 된다. 따라서, 데이터의 에러를 방지하게 된다. 이 외에 현재 데이터를 읽고 있거나 쓰고 있는 도중에 데이터의 언더플로우(underflow)를 방지하기 위한 프로세스가 있다. 이는 데이터 입출력 동작 도중에, 입력 버퍼의 읽기/쓰기(read/write) 어드레스의 차이가 '1'만큼 날 경우에 언더플로우(underflow) 신호를 생성시켜 준다. 언더플로우(underflow) 신호는 에스디램(SDRAM)의 제어신호를 생성하는 부분으로 출력되어 데이터 쓰기(write) 동작을 중지시키게 된다.

(2) 출력 인터페이스장치

메모리에 저장된 데이터들은 항등률의 데이터 비트율을 가진 외부 채널로 출력된다. 데이터의 출력은 출력 인터페이스장치(200)를 통하여 이루어지는데, 이것의 기능은 전술한 입력 인터페이스장치(100)와 근본적으로는 비슷하다. 외부에서 채널로 데이터를 출력하라는 신호는 cstrm_en 이라는 인에이블 신호를 통하여 전달된다. 이 신호가 '1'인 동안에 데이터를 외부로 출력하게 되는데, 출력 버퍼(800)에 저장되어 있는 32 비트 데이터를 8 비트씩 나누어서 출력해 주어야 한다. 이러한 기능을 구현하기 위해서 기존에는 8개의 에스램(SRAM)을 사용했지만, 이럴 경우에 면적과 버퍼 사용에 있어서 너무 비효율적이 된다. 따라서, 한 개의 에스디램(SDRAM) 버퍼를 사용하고 32 비트의 데이터를 플립플롭(F/F)으로 잡아서 출력시키는 방법을 사용하였다. 또한, 유효한 데이터를 안정적으로 출력시키기 위해서 입력되는 인에이블 신호보다 한 클럭 지연시켜서 데이터를 출력시키도록 설계하였다. 메모리에서 데이터를 읽어올 때는 입력 인터페이스장치(100)와 마찬가지로 제어신호 생성기(300)로부터 인에이블 신호를 받는다. 메모리 액세스가 허가되는 타이밍 동안에 54㎒에 동기하여 데이터를 입력 받는다. 입력은 32 비트로 이루어지므로 컨트롤(control)에 어려움은 없다. 여기서, 출력되는 데이터는 27㎒로 움직이고 입력 데이터는 54㎒로 동기되어 있기 때문에 데이터의 오버플로우(overflow)가 발생할 수 있다. 따라서, OUT_SDRAM_SPACE 라는 플래그(flag) 신호를 사용하였다. 앞에서와 마찬가지로 입력되는 데이터와 출력되는 데이터의 어드레스 차이가 '1'이 되는 때에 이 신호를 '1'로 세팅(setting)시켜서 제어신호 생성기(300)로 출력해 준다. 이 신호가 발생하면 제어신호 생성기는 적절한 조치를 취하게 된다. out_proh 신호는 입력 인터페이스장치(100)의 in_proh 신호와 그 역할이 비슷하다. out_proh 신호는 에스디램(600) 입출력 데이터의 어드레스에 따라 같은 경우 '1'의 값이 출력된다. 이 신호는 제어신호 생성기(300)로 출력되고 에스디램(600) 어드레스 제어 신호의 출력을 조절한다. 즉, 에스디램(600)에서 잘못된 데이터 값을 읽어오지 않도록 방지하는 기능을 수행한다.

도 6은 출력 인터페이스장치(200)의 입출력 구조를 나타낸 것이며, 주요 핀을 설명하면 다음과 같다.

cstrm_eb : 외부 채널로 데이터를 출력하라는 인에이블 신호.

SDRAMR_p_incr : 메모리로부터 데이터를 입력받아 버퍼에 저장하도록 하는 인에이블 신호.

out_proh : 입출력 데이터의 어드레스를 비교하여 오버리드(overead)를 방지하는 신호.

OUT_SRAM_SPACE : 에스램(SRAM) 데이터의 오버플로우(overflow)를 막기 위한 플래그(flag) 신호.

도7은 출력 인터페이스장치(200)의 내부 구조이다. 채널로 출력되는 데이터는 외부에서 입력되는 cstrm_en 신호에 의해 이루어진다. 이 신호가 '1'인 동안에 에스램(SRAM)의 읽기(read) 어드레스가 증가하면서 버퍼에 어드레스를 주고, 이 어드레스에 따라서 각각의 데이터는 플립플롭(F/F)으로 출력된다. 이 데이터들은 여기서 다시 다중화(MUX)되어 8 비트씩 채널로 출력된다. 읽기(read) 어드레스를 나타내는 카운터는 0-15 의 값을 갖는 4 비트면 충분하다. 데이터를 다중화(MUX)하는 신호는 cstrm_en 과 sck2 신호를 이용하여 생성한다. 여기에는 별도로 2비트의 카운터를 두어서 32 비트를 각각 8 비트로 나눌 때 기준으로 사용한다. 따라서, 이 카운터의 값이 "11"이 될 때마다 에스디램(SDRAM)의 데이터를 새로 읽어내도록 설계하였다. 즉, 에스디램 읽기(SDRAM read) 어드레스는 이 카운터의 값이 '3'이 될 때마다 하나씩 증가한다. 메모리에서 데이터를 읽어낼 때는 제어신호 생성기로부터 인에이블 신호를 입력받는다. SDRAMR_p_incr 신호가 '0'이면 이 때가 바로 부호화 장치로부터 메모리 액세스를 허가받은 타이밍이므로 에스디램 쓰기(SRAM write) 어드레스 카운터를 증가시킨다. SRAMWE 신호는 sck4와 SDRAMR_pp_incr 신호를 '1'에서 '0'으로 떨어지는 폴링 에지(falling edge)에서 유효한 데이터가 입력되므로 어드레스가 이때 유효하도록 만들어 줘야 한다. 입력은 54㎒로 동기되고, 출력은 27㎒로 동기되기 때문에, 자칫 데이터의 오버플로우(overflow)가 발생할 수 있다. 이를 방지하기 위한 것이 바로 OUT_STAM_SPACE이다. 원리는 입력 버퍼에서 사용한 pre_zero_detect와 유사하다. 이 신호가 '1'이 되면 제어신호 생성(Control signal generator) 모듈이 SDRAMR_p_incr 신호를 '1'로 만들어 주고, 이 때문에 더 이상 에스램(SRAM)은 데이터를 입력받을 수 없도록 설계하였다.

도 8은 출력 버퍼의 어드레스 포이너를 구조화하여 나타낸 것이고, 도 9는 출력 인터페이스 장치의 입출력 데이터 타이밍을 나타낸 것이다. 먼저, 출력 버퍼의 입출력 어드레스를 생성하는 부분은 입력 인터페이스장치(100)의 부분과 비슷하다. 그러나, 여기에서 다른 점은 에스디램(600)에서 출력되는 데이터의 지연을 고려해야 한다는 점이다. sram_write_addr를 세는 카운터는 delay4_enable 신호에 의해 인에이블된다. delat4_enable 신호는 실제로 에스디램(SDRAM)에서 데이터가 지연되어 출력되는 타이밍을 나타낸다. real_sramwe_cnt는 데이터의 오버플로우(overflow)를 방지하기 위해서 에스램(SRAM)의 어드레스를 비교하기 위해 사용한다. sram_we_make 신호는 실제로 데이터가 입력되는 타이밍을 나타내는 delay3_enable 신호가 '0'일 때마다 반전되는 인에이블 신호이다. 이 값을 반클럭 만큼 지연시켜서 논리 회로로 조합하여 sram_we 신호를 생성시켜 주었다. 출력 버퍼(800)는 이 신호와 입력 어드레스를 입력받아서 데이터를 저장하게 된다. 데이터의 출력은 출력 어드레스를 버퍼에 인가함으로써 이루어진다. 출력 어드레스는 입력 어드레스보다 조금 더 복잡하다. 왜냐하면, 채널의 출력은 바이트 단위로 이루어지는 반면, 버퍼의 출력은 32비트 단위이기 때문이다. four2one_mux는 외부 채널로 출력되는 데이터의 인덱스를 나타낸다. 32 비트 데이터가 8 비트씩 출력되기 때문에 현재 출력되는 데이터가 32 비트 중에서 어느 바이트인지를 나타내주는 2 비트 카운터이다. 출력되는 데이터는 32 비트씩 출력축에 있는 플립플롭(F/F)에 저장되어 있다가 이 값에 따라서 인에이블신호가 입력되면 외부 채널로 출력된다. 또한, 이 값을 참조하여 새롭게 플립플롭(F/F)의 데이터값을 갱신(update)시켜야 하는지를 결정하게 된다. sramread_en 신호는 실제로 출력 버퍼에서 데이터를 플립플롭(F/F)으로 읽어내는 타이밍을 나타내는 인에이블 신호이다. 이 신호는 앞에서 살펴본 four2one_mux 값이 "0"일때마다 '1'이 된다. 이 값이 '1'이고 외부 채널의 인에이블 신호가 '1'일 때마다 출력 버퍼의 출력 어드레스값이 '1'씩 증가하게 된다. 증가된 값은 바로 출력 버퍼로 연결되어 다음 어드레스의 데이터가 출력되고 이 값이 플립플롭(F/F)의 데이터를 갱신(update)시켜 준다. 이 외에도 출력 버퍼의 오버플로우(overflow)를 막기 위한 프로세스가 별도로 추가되어 있다. out_proh 신호는 출력 버퍼의 어드레스를 비교하여 두 어드레스가 같게 되는 경우에 즉, 외부 채널의 데이터 요구량이 적은 경우에 에스디램(SDRAM)으로부터의 데이터 출력을 억제하는 역할을 한다. 이 신호가 '1'이 되면 에스디램(SDRAM)의 제어신호가 변경되어 데이터 액세스를 하지 않도록 해 준다. 또한, 이 경우 외에도 데이터의

오버플로우(overflow)가 발생될 수 있다. 현재 데이터를 에스램(SRAM)에서 읽거나 쓰고 있을 경우에는 아래와 같은 프로세스를 이용하여 오버플로우(overflow)를 방지할 수 있다. 출력 버퍼(800)의 읽기/쓰기(read/write) 어드레스를 비교하여 그 차이가 '1'이 되면 오버플로우(overflow) 신호를 '1'로 세팅시켜 준다. 이 신호가 에스디램(SDRAM)의 제어 신호를 생성시키는 부분으로 출력되면 데이터 액세스 도중에 미리 데이터 읽기(read)를 하지 못하도록 막을 수 있다.

(3) 제어신호 생성기

메모리의 제어 신호와 어드레스는 물론, 입력 버퍼와 출력 버퍼의 데이터 입출력 인에이블 신호를 생성시켜 주는 모듈(module)이 제어신호 생성기(300)이다. 전체적인 설계는 부호화 장치로부터 입력받는 인에이블 신호에 따라서 이루어졌다. 각각 메모리에 데이터를 입력하거나 메모리로부터 데이터를 출력하는 타이밍이 되면 그에 따른 신호들을 만들어 주는 유한 상태기(FSM : Finite State Machine)를 구성하였다. 입력과 출력을 위하여 각각 카운터를 사용하였고, 두 가지 경우에 생성되는 신호들을 다중화(MUX)하기 위한 로직을 구현하였다. 특별히 2-뱅크 동작(BANK OPERATION)을 실현하기 위한 로직이 구성되었고, 오버플로우(overflow)나 언더플로우(underflow)가 발생하는 경우를 방지하기 위한 로직을 추가하였다.

도 10은 제어신호 생성기의 입출력 구성을 나타낸 것이며, 주요 핀들을 설명하면 다음과 같다.

V2F_en : 메모리에 write 하는 타이밍을 나타내는 인에이블 신호.

F2C_en : 메모리에 read 하는 타이밍을 나타내는 인에이블 신호.

OUT_SRAM_SPACE : 출력 버퍼(800)의 overflow를 나타내는 flag 신호.

pre_zero_detect : 입력 버퍼(700)의 underflow를 나타내는 flag 신호.

row_deac : 에스디램(SDRAM)의 어드레스가 뱅크를 변환시킬 때 발생하는 플래그(flag) 신호.

VLCSelectV2F,VLCSelectF2C : 입출력 어드레스와 데이터, 제어신호를 구별하기 위한 신호.

RADDR,BANK,CADDR : 에스디램(SDRAM) 어드레스 생성기(400)로부터 입력받는 메모리어드레스.

RAS,B,CAS,WE,CS,DQM ; 메모리 제어신호.

ADDR(9:0) : 메모리로 출력되는 입출력 어드레스.

SDRAMR_p_incr : 출력 버퍼(800)로 출력 되는 데이터의 유효성을 나타내는 인에이블 신호.

WE_SDRAM : 입력 버퍼(700)로 출력 되는 데이터의 유효성을 나타내는 인에이블 신호.

in_proh : 에스디램(600)이 잘못된 데이터를 쓰기(write)하지 못하도록 방지하는 플래그(flag) 신호.

out_proh : 에스디램(SDRAM)이 잘못된 데이터를 읽지 못하도록 방지하는 플래그(flag) 신호.

도 11은 도 10의 내부 구조를 나타낸 것이다. 부호화 장치가 V2F_en 신호를 '1'로 만들면 메모리에 데이터를 쓰기(write)해도 된다는 허가를 내린 경우에, V2F_en 신호와 sck4클럭을 이용하여 에스디램(SDRAM)에 데이터를 쓰는 타이밍을 계산하는 카운터를 증가시킨다. 이 값은 0-52까지 증가하기 때문에 6비트 카운터를 쓰면 충분하다. 이 카운터값을 이용하여 스텝(step)을 계산한다. 한편으로는, V2F_en 두 신호와 sck4 클럭을 이용하여 스텝(step)의 각 단계별로 해당하는 제어신호를 생성하여 메모리로 출력해 준다. 정상적인 경우에는 카운터값이 52가 되면 타이밍이 끝나지만, 만약 입력 버퍼에 언더플로우(underflow)가 발생하거나, 쓰다가 뱅크(BANK)를 바꿔야 하는 경우 등이 발생하면, 이에 따라 다른 제어신호를 생성하여 출력시켜 주는 로직이 추가되어 있다. 반대의 경우로 F2C_en 신호가 '1'이 되면, 이 때는 메모리로부터 데이터를 읽어내게 된다. 이 과정은 메모리에 데이터를 쓰는 것과 크게 다르지 않다. F2C_en 신호와 sck4 클럭을 이용하여 읽기(read) 카운터를 증가시키고, 이 값을 입력으로 받아서 스텝(step)에 따른 제어신호를 생성하게 된다. 메모리에서 데이터를 읽어내는데 할당된 타이밍은 0-9까지 이므로 이 카운터는 4 비트의 크기를 가지게 된다. 역시 정상적인 경우에는 카운터값이 9가 될 때까지 증가하지만, 출력 버퍼에서 오버플로우(overflow)가 발생하거나 뱅크 동작(BANK OPERATION)을 해야 하는 경우에는 이에 따른 다른 제어신호가 생성되도록 되어 있다. 이렇게 생성된 제어 신호는 에스디램(SDRAM) 어드레스 생성기로부터 입력된 데이터 어드레스와 함께 메모리로 출력되는데, VLCSelectV2F 신호에 의해서 다중화(MUX)되어 출력된다. VLCSelectV2F 신호가 '0'이면 데이터를 쓰기(write)하고 반대로 '1'이면 데이터를 읽기(read)하는 타이밍으로 설정되어 있다.

도 12 의 타이밍도에서 이와 같은 과정을 간단하게 나타내었다. 먼저, 에스디램(SDRAM)의 사용 허가를 나타내는 V2F_en 신호와 F2C_en 신호가 있을 때마다 읽기/쓰기(read/write) 카운터를 이용하여 각각의 스텝(step)을 결정한다. 각각 읽기/쓰기(read/write)별로 할당된 타이밍에 따라서 스텝(step)이 결정된다. 쓰기(write)의 경우에는 53개, 읽기(read)의 경우에는 10개씩의 클럭이 할당된다. 설계된 구조에서는 한 번에 몇 개의 데이터가 입출력 될지를 모르기 때문에, 데이터 입출력에 필요한 클럭 수를 줄이기 위해서 데이터의 입출력시에는 항상 두 개의 뱅크를 모두 활성화(activate)시켜서 사용하고 있다. 따라서, 매번 스텝(step)의 처음에는 두 뱅크를 활성화(activate)시켜주는 부분이 포함되어 있다. 이렇게 결정된 스텝(step)에 따라 각각의 제어신호를 생성하게 된다. 먼저, cs 신호의 경우를 살펴보면, V2F_en,F2C_en이 '1'인 경우에는 '0'이고 나머지 경우에는 '1'을 출력시켜서 메모리를 선택하도록 설계하였다. 두 번째로는 we 신호를 살펴보면, 쓰기(write)의 경우, write_timestep의 처음 두 개(뱅크 row_activate)와 no_operation(데이터의 쓰기 중간에 언더플로우가 발생한 경우)의 값은 '1'이고 나머지는 모두 '0'으로 데이터의 쓰기(write)가 가능하도록 되어 있다. 읽기(read)의 경우에는 we 값이 '1'이면 읽기(read) 가능한 경우이므로 '0'이 되는 경우만을 설명하면 데이터를 읽어내다가 뱅크가 바뀌는 경우에는 읽기(read)를 중단해야 하므로 '0'이 되도록 하였고, 읽어내는 도중에 출력 버퍼에서 데이터 오버플로우(overflow)가 발생하는 경우도 '0'이 되도록 하였다. 또한, 두 개의 뱅크를 항상 활성화(activate)시켜 놓기 때문에 맨 마지막 스텝(step)에서는 주의를 해 주어야 한다. 뱅크를 바꾸었을 경우에는 마지막 한 개만이 로우_비활성화(row_deactivate) 신호이지만, 그렇지 않을 경우에는 두 개의 스텝(step)을 로우_비활성화

(row_deactivate) 신호로 만들어 주어야 하기 때문에 이와 같은 경우에는 '1'을 만들어 주었다. dqm 신호에 대해서는, dqm 신호가 '1'이면 데이터가 마스킹(masking)되어 입출력이 되지 않는다. 따라서, 쓰기(write)를 못하도록 하는 역할을 하는데, 뱅크를 바꾸는 경우에는 '1'로 만들어서 이 같은 역할을 하도록 하였다. 읽기(read)의 경우에는 항상 '0'이 되어도 상관이 없기 때문에 간편하게 '0'의 값을 인가하였다. ras 신호는 로우_활성화(row_activate)와 로우_비활성화(row_deactivate)에 중요한 역할을 한다. 데이터를 쓰기(write)하는 경우에 로우_활성화(row_activate)와 로우_비활성화(row_deactivate)시에는 '0'의 값을 갖게 된다. 그 이외에는 데이터를 쓰기(write)하는 도중에 입력 버퍼의 데이터 언더플로우(underflow)가 발생할 경우에 '0'의 값을 갖게 되고, 중간에 뱅크를 바꿨는지에 따라서 마지막 두 개의 스텝(step)에서 갖게 되는 값이 달라질 수 있다. 실제로 데이터를 쓰기(write)할 때는 '1'의 값을 가지게 된다. 읽기(read)의 경우에도 비슷한 값을 갖게 된다. 로우_활성화(row_activate)와 로우_비활성화(row_deactivate)의 경우, 그리고 읽기(read)하는 도중에 데이터의 오버플로우(overflow)가 발생할 경우에는 '0'의 값을 갖게 된다. 그 이외에는 '1'의 값을 가지게 된다. cas 신호의 경우에는 보통 에스디램(SDRAM)을 사용하는 경우와 조금 다르다. 보통 디램(DRAM)의 경우에는 데이터를 읽거나 쓸 때마다 cas 신호를 만들어 주어야 하지만, 에스디램(SDRAM)을 사용할 경우에는 데이터의 시작 부분에서만 cas 신호를 만들어 주면 그 다음부터는 에스디램(SDRAM) 내부의 카운터에 의해 자동으로 데이터의 입출력이 이루어지기 때문에 더욱 간편하게 구현할 수 있다. 먼저, 쓰기(write)의 경우에는, 로우_활성화(row_activate)와 로우_비활성화(row_deactivate)시켜주는 부분인 스텝(step)의 처음 두 개, 그리고 마지막의 두 개 부분에서 cas 신호를 '1'로 만들어 준다. 마지막의 두 신호를 만들 때는 뱅크의 전환 여부에 따라서 데이터를 쓰는 신호를 만들어 주거나 로우_비활성화(row_deactivate) 신호를 만들어 주고 비동작(no_operation)을 수행하는 제어 신호를 발생시켜 준다. 데이터를 쓰는 경우에는 처음 시작하는 데이터에서만 cas 신호를 '0'으로 만들어 주고 나머지는 '1'로 유지시켜 준다. 이 부분이 보통의 디램(DRAM)과 다른 부분이다. 읽기(read)의 경우도 쓰기(write)의 경우와 비슷하다. 읽기(read)의 경우에는 출력 버퍼에서 오버플로우(overflow)가 발생하는 경우에만 별도로 cas 신호를 생성시켜 주고 있다. 실제로 데이터를 읽기(read)하는 부분을 제외한 부분은 모두 '1'의 값을 주고, 읽기(read) 데이터의 처음 부분만 '0'으로 만들어서 데이터 읽기(read)를 알려 주도록 하였다. 뱅크(bank) 신호의 경우는 에스디램(SDRAM) 어드레스 생성기에서 뱅크신호를 입력받는다. 그러나, 인위적으로 뱅크를 바꾸어야 하는 경우가 있다. 스텝(step)의 처음 두 개 신호는 각 뱅크의 로우_비활성화(row_activate) 신호이다. 따라서, 현재의 뱅크와 다음 뱅크를 차례로 활성화(activate)시키기 위해서 두 번째 경우에는 뱅크를 인위적으로 바꾸어 준다. 또한, 쓰기(write) 도중에 비동작(no_operation)이 발생하는 경우나 마지막 스텝(step)에서 로우_비활성화(row_deactivate)를 해 줘야 하는 경우에도 뱅크를 인위적으로 반전시켜 준다. 읽기(read)의 경우도 마찬가지로 구현할 수 있으나 단지 다른 점은 여기에서의 비동작(no_operation)은 출력 버퍼의 오버플로우(overflow)로 인해 발생한다는 점이다. 실제로 데이터의 읽기/쓰기(read/write)가 발생하는 시점을 나타내는 인에이블 신호 생성은 에스디램(SDRAM) 어드레스 생성기와 출력 인터페이스 장치로 출력되어 에스디램(SDRAM)의 어드레스 생성과 입출력 데이터의 개수 조절, 입출력 데이터의 쓰기 인에이블(write_enable) 신호 생성에 사용된다. 앞에서 설명한 로우_활성화(row_activate)와 로우_비활성화(row_deactivate), 오버플로우(overflow), 언더플로우(underflow) 등과 같이 데이터의 입출력이 아닌 경우에는 모두 '1'의 값이 출력되고 실제 데이터의 이동이 있는 경우에는 '0'의 값이 출력되어 인에이블 신호로서 작용하게 된다.

다음으로는 에스디램의 어드레스를 생성하는 부분을 살펴보도록 하겠다. 어드레스는 에스디램(SDRAM) 어드레스 생성기로부터 입력되는데 이 값이 그대로 출력되는 것이 아니라, 읽기/쓰기(read/write)에 따라서 다중화(MUX)되어 출력된다. 또한, 설계된 구조에서는 항상 두 개의 뱅크를 모두 활성화(activate)시켜서 사용하므로 로우(row) 어드레스의 출력에서는 조금 주의해야 할 부분이 있다. 현재의 동작이 읽기(read)인지 쓰기(write)인지에 따라서 일단 어드레스를 나눈다. 그리고 난 후에는, 현재의 뱅크가 어느 뱅크인지에 따라서 다시 로우(row) 어드레스가 달라진다. 그 이유는 항상 두 개의 뱅크를 선택하기 때문이다. 처음 두 개의 스텝(step)에서 두 개의 뱅크를 활성화(activate)시키는데, 만약 현재의 뱅크가 '1' 뱅크라면 '0'뱅크를 활성화(activate)시키기 위해서는 현재의 로우 어드레스를 하나만큼 증가시켜야 하기 때문이다. 일단 로우_활성화(row_activate)가 되고 나면, 나머지는 칼럼(column) 어드레스가 출력되도록 하였다. 이렇게 생성된 제어신호들은 앞에서 설명한 in_proh, out_proh 신호에 의해 마스킹(masking)이 되어 사용된다. 생성된 신호들은 각각 write_control_sig 와 read_control_sig로 전달되는데, 이 신호들은 정상적인 경우에는 그대로 외부로 출력되지만, 만약 현재 데이터를 읽거나 써야 하는 경우에 오버플로우(overflow)나 언더플로우(underflow)가 발생하면 비동작(no_operation : 코드에서 "1110")값을 출력시키도록 하고 있다. 또한, 실제 데이터의 입출력을 나타내는 SDRAM_p_incr 신호와 WE_SDRAM 신호도 in_proh 신호와 out_proh 신호와의 논리합(OR) 연산에 의해 마스킹(masking)이 되고 있다. 이렇게 함으로써 정확한 데이터의 입출력을 보장할 수 있다.

(4) 에스디램(SDRAM) 어드레스 생성기

에스디램(SDRAM) 어드레스 생성기(400)는 메모리로부터 데이터를 읽거나 쓸 때의 어드레스를 생성하여 출력시켜 주는 장치이다. 메모리의 제어 신호를 생성하는 부분과는 별도로 어드레스 생성기(400)를 둠으로써 간단하게 어드레스를 만들 수 있다. 어드레스를 나타내는 데에는 로우(row) 어드레스와 뱅크(BANK), 칼럼(column) 어드레스를 나타내는 카운터 포인터를 이용한다.

도 13은 에스디램(SDRAM) 어드레스 생성기(400)의 어드레스 포인터 구조를 나타낸 것이다. 인에이블 신호는 제어신호 생성기로부터 입력 받는다. 데이터를 읽을 때는 SDRAMR_p_incr 신호를 입력 받고, 반대로 데이터를 쓸 때는 WE_SDRAM 신호를 입력 받는다. 두 경우에 모두 인에이블 신호가 '0'일때가 액티브 상태가 된다. 메모리에서 데이터를 읽기(read)하거나 쓰기(write)하는 경우를 각각 나눠서 다른 모듈을 사용하도록 설계하였다. 서로 같은 구조의 모듈 두 개로 각각의 경우에 어드레스를 계산하여 출력할 때는 VLCSelectV2F 신호와 VLCSelectF2C 신호를 이용하여 다중화(MUX)하는 구조를 채택하였다.

도 14는 에스디램 어드레스 생성기의 입출력 구성을 나타낸 것이며, 주요 핀을 설명하면 다음과 같다.

ENABLE : 데이터를 읽거나 쓰도록 허가해주는 인에이블 신호.

RADDR : 로우 어드레스.

BANK : 뱅크를 선택하는 신호.

CADDR : 칼럼(column) 어드레스.

row_deac. : 뱅크 동작(BANK OPERATION)에서 뱅크를 바꿀 때 생성하는 신호.

칼럼(column) 어드레스는 0-179 까지 증가하고, 로우(row) 어드레스는 660-1023 까지 증가한다. 인에이블 신호가 '0'인 동안에 칼럼 어드레스가 '1'씩 sck4에 동기되어 증가한다. 그러다가 칼럼 어드레스가 179가 됐을 때, 인에이블 신호가 '0'이라면 뱅크(BANK)가 토글(toggle)되면서 로우 어드레스도 '1'만큼 증가한다. 또한, 뱅크 동작(BANK OPERATION)을 하기 위한 수단도 포함되어 있다. 도 15는 에스디램(SDRAM) 어드레스 생성기의 내부 구조를 나타낸 것이다. 어드레스가 안정적으로 변하도록 하고, 유효한, 어드레스를 얻기 위해 각 신호들을 플립플롭(F/F)을 이용하여 반 클럭씩 지연시켜서 사용한다.

실제로 어드레스를 계산하는 과정을 도 16의 타이밍도에 나타내었다. 도 16에서 보면, 칼럼(column) 어드레스가 179가 되면 뱅크를 바꾸어 줘야 한다. 뱅크 전환을 위해서는 먼저 현재 사용하고 있던 뱅크를 로우_비활성화(row_deactivate) 시켜줘야 하는데, out_row_deac 신호가 '1'로 세팅이 되면 뱅크가 전환되면서 로우 어드레스가 바뀌게 된다. row_deac 신호는 뱅크를 반전시켜 주고, 이 뱅크 신호가 '1'에서 '0'으로 떨어지는 플립 에지에서만 로우 어드레스가 '1'만큼 증가한다. 로우 어드레스는 '660'에서 '1023'의 값을 가지게 된다.

(5) 버퍼 상태 연산기

비트스트림의 플럭률 비트율에 따라서 항등률로 출력하기 위해서는 현재 채널 버퍼에 저장되어 있는 데이터의 양을 알아야 하고, 이 정보를 가지고 가변길이 부호화의 속도를 조절해야 한다. 따라서, VLC 서브모듈에서는 가변길이 부호화의 속도를 결정하는 버퍼의 상태를 출력시켜 주는 기능을 가지고 있다. 이와 같은 가능을 하는 것이 버퍼 상태 연산기이다.

도 17은 버퍼 상태 연산기의 입출력 구성을 나타낸 것이며, 주요 핀을 살펴보면 다음과 같다.

vstrm_en : 데이터의 입력을 나타내는 인에이블 신호.

cstrm_en : 데이터의 출력을 나타내는 인에이블 신호.

bstrmp_en : 버퍼의 상태를 출력하라는 인에이블 신호.

BSTAE : 버퍼의 상태(바이트 단위).

버퍼에 저장된 데이터의 개수는 입력된 데이터의 개수와 출력된 데이터의 개수의 차이를 계산하면 알 수 있다. 이를 위하여 입력 인터페이스 장치(100)와 출력 인터페이스 장치(200)로 입출력되는 데이터의 개수를 알아야 하는데, 데이터의 개수는 입출력되는 데이터의 인에이블 신호를 이용하여 계산한다. 즉, 인에이블 신호가 '1'이 될 때마다 카운터값을 증가시켜서 입출력되는 데이터의 개수를 계산하는 방법을 사용하였다. 그런데, 입력 버퍼(700)로 들어오는 데이터는 32비트 단위이고 출력 버퍼를 통해 나가는 데이터는 8 비트이기 때문에 입력 버퍼로 들어오는 데이터의 개수에 4를 곱해 줌으로써 8 비트 단위의 데이터 개수를 구할 수 있다.

도 18은 버퍼 상태 연산기의 내부 구조이며, 도 19는 동작 타이밍도이다. 한 매크로블록 동안에 입력되는 데이터(최종 부호화된 데이터의) 개수는 최대 291 개다. 따라서, 입력 데이터의 개수는 vstrm_en 이 될 때마다 증가하는 9비트 카운터를 이용하여 계산하여 vstrm_count 값으로 저장한다. 이 값은 매번 버퍼의 상태를 계산할 때마다(bstmp_en의 라이징 에지에서) pre_vstrm_count 에 저장된다. 실제로 버퍼 상태 계산에 사용되는 값은 이 두 값의 차이로서 매 매크로블록 마다 증가된 입력 데이터의 개수를 계산하여 출력시켜 준다. 버퍼의 상태는 bstmp_en 의 라이징 에지에서 계산되는데, 앞에서 구해진 매크로블록 마다의 입출력 데이터의 증감을 입력으로 받아서 이전에 계산된 버퍼의 상태에 새롭게 계산된 버퍼의 상태를 더해서 계산한다. 버퍼의 상태를 계산하는 인에이블 신호는 매 매크로블록 출력마다 1258번째 클럭에서 세팅된다. 설계된 모듈의 구조상 입출력 데이터의 개수는 단순히 입력 데이터의 인에이블 신호와 출력 데이터의 요구 신호를 이용하면 구할 수 있다. 입력되는 데이터의 개수는 vstrm_en 이 '1'인 동안에 증가하는 카운터를 이용하여 계산한다. 한 매크로블록 동안에 입력되는 데이터의 최대 개수가 291개이므로 291개를 세는 9비트 카운터를 이용하였다. 출력되는 데이터의 개수는 cstrm_en 이 '1'인 동안에 '1'씩 증가하는 7비트 카운터를 이용하여 계산하였다. 이렇게 계산된 입출력 데이터의 개수는 현재 매크로블록에서의 버퍼 상태를 계산한 직후에 각각 pre_vstrm_count와 pre_cstrm_count에 저장된다. 따라서, 매 매크로블록마다 실제로 계산되는 데이터의 개수는 이전 매크로블록까지 입출력된 데이터의 개수와 현재 매크로블록까지 입출력된 데이터의 개수의 차이, 즉 한 매크로블록 동안의 데이터 입출력 개수만을 계산하게 된다. 매번 버퍼의 상태를 계산하라는 인에이블 신호(bstate_en)의 플링 에지에서 현재의 값을 과거의 값으로 저장하여 갱신(update)하게 된다. 여기서, vstrm_check와 cstrm_check가 '1'인 경우에는 저장하는 값에서 '1'씩 빼게 된다. vstrm_check와 cstrm_check 신호는 현재 버퍼의 상태를 계산하는 도중에 데이터의 입출력이 있었다면 '1'로 세팅되는 신호들이다. 따라서, 이 신호들이 '1'인 경우에는 과거의 값으로 저장하는 데이터의 수에서 '1'만큼을 뺌으로써 정확한 데이터의 개수를 알 수 있다. 버퍼의 상태를 계산하라는 신호인 bstate_en 이 '1'일 때, vstrm_en 이나 cstrm_en 이 '1'인 경우에 이 신호들을 '1'로 만들어 주면 된다. 각각 bstate_en 신호의 라이징 에지에서, 이전까지 증가한 데이터의 카운트값과 현재 데이터의 카운터값의 차이를 계산한다. 이 값들은 bstate_en 의 플링 에지에서 버퍼의 크기를 계산하여 출력하는데 사용된다. 버퍼의 크기는 bstate_en의 플링 에지에서 계산된다. 이전까지의 버퍼의 크기를 나타내는 pre_buffer_size와 현재 계산된 데이터의 개수를 더해서 계산한다. 현재의 데이터 개수는 위에서 보는 것과 같이, vstrm의 개수에 4를 곱한 후, cstrm의 개수를 빼서 구한다. 이렇게 하는 이유는 vstrm은 32비트 단위로 입력되고 cstrm은 8비트 단위로 출력되기 때문이다. 따라서, 계산된 데이터의 개수는 바이트단위를 갖게 된다. pre_buffer_size는 버퍼의 크기를 계산하기 직전에 이전의 값으로 갱신(update)된다.

이상에서 설명한 본 발명은 전술한 실시예 및 첨부된 도면에 의해 한정되는 것이 아니고, 본 발명의 기술적 사상을 벗어나지 않는 범위 내에서 여러 가지 치환, 변형 및 변경이 가능하다는 것이 본 발명이 속하

는 기술분야에서 통상의 지식을 가진 자에게 있어 명백할 것이다.

발명의 효과

이상에서 설명한 바와 같이 본 발명에서 제시하는 부호화 영상 데이터의 인터페이스 장치는 기존의 속도가 느린 비동기식 디램을 사용하지 않고 속도가 빠르고 제어가 간편한 동기식 에스디램(SDRAM)을 사용하여 인터페이스 장치내의 비트폭과 하드웨어 복잡도를 줄여 하드웨어 구현시 비용을 줄였다. 이는 비디오 인코더 칩의 가격을 저렴하게 하고 전력 소모를 줄이는 효과가 있다.

(57) 청구의 범위

청구항 1

부호화 영상데이터의 인터페이스장치에 있어서,

어드레스 지정에 따라 데이터의 입/출력이 이루어지는 에스디램;

상기 에스디램의 데이터 리드 및 라이트를 위한 어드레스를 발생하는 어드레스 생성 수단;

부호화 장치로부터 입력된 데이터를 저장하여 상기 에스디램으로 출력하는 입력 인터페이싱 수단;

상기 에스디램에 저장된 데이터를 외부채널로 출력하는 출력 인터페이싱 수단; 및

상기 에스디램의 데이터 입력과 출력을 제어하는 제어신호 생성 수단

을 포함하는 부호화 영상데이터의 인터페이스장치.

청구항 2

제 1 항에 있어서,

상기 입력 인터페이싱 수단과 입력 및 출력이 이루어지는 입력 버퍼링 수단

을 더 포함하는 부호화 영상데이터의 인터페이스장치.

청구항 3

제 2 항에 있어서,

상기 입력 버퍼링 수단이 에스램(SRAM)으로 이루어진 것을 특징으로 하는 부호화 영상데이터의 인터페이스장치.

청구항 4

제 1 항에 있어서,

상기 출력 인터페이스 수단과 입력 및 출력이 이루어지는 출력 버퍼링 수단

을 더 포함하는 부호화 영상데이터의 인터페이스장치.

청구항 5

제 4 항에 있어서,

상기 출력 버퍼링 수단이 에스램(SRAM)으로 이루어진 것을 특징으로 하는 부호화 영상데이터의 인터페이스장치.

청구항 6

제 1 항에 있어서,

상기 어드레스 생성 수단은,

읽기 어드레스 생성기와 쓰기 어드레스 생성기로 이루어진 것을 특징으로 하는 부호화 영상데이터의 인터페이스장치.

청구항 7

제 1 항에 있어서,

상기 어드레스 생성 수단과 상기 제어신호 생성 수단 사이에 멀티플렉서를 더 포함하는 것을 특징으로 하는 부호화 영상데이터의 인터페이스장치.

청구항 8

부호화 영상데이터의 인터페이스장치에 있어서,

어드레스 지정에 따라 데이터의 입/출력이 이루어지는 에스디램;

상기 에스디램의 데이터 리드 및 라이트를 위한 어드레스를 발생하는 어드레스 생성 수단;

부호화 장치로부터 입력된 데이터를 저장하여 상기 에스디램으로 출력하는 입력 인터페싱 수단;

상기 에스디램에 저장된 데이터를 외부채널로 출력하는 출력 인터페이싱 수단;

상기 에스디램의 데이터 입력과 출력을 제어하는 제어신호 생성 수단; 및

현재의 채널버퍼에 저장된 데이터의 정보를 이용하여 가변길이 부호화의 속도를 결정하는 버퍼상태 연산 수단

을 포함하는 부호화 영상데이터의 인터페이스장치.

청구항 9

제 8 항에 있어서,

상기 입력 인터페이싱 수단과 입력 및 출력이 이루어지는 입력 버퍼링 수단

을 더 포함하는 부호화 영상데이터의 인터페이스장치.

청구항 10

제 9 항에 있어서,

상기 입력 버퍼링 수단은 에스램(SRAM)으로 이루어진 것을 특징으로 하는 부호화 영상데이터의 인터페이스장치.

청구항 11

제 8 항에 있어서,

상기 출력 인터페이싱 수단과 입력 및 출력이 이루어지는 출력 버퍼링 수단

을 더 포함하는 부호화 영상데이터의 인터페이스장치.

청구항 12

제 11 항에 있어서,

상기 출력 버퍼링 수단이 에스램(SRAM)으로 이루어진 것을 특징으로 하는 부호화 영상데이터의 인터페이스장치.

청구항 13

제 8 항에 있어서,

상기 어드레스 생성 수단은,

읽기 어드레스 생성기와 쓰기 어드레스 생성기로 이루어진 것을 특징으로 하는 부호화 영상데이터의 인터페이스장치.

청구항 14

제 8 항에 있어서,

상기 어드레스 생성 수단과 상기 제어신호 생성 수단 사이에 멀티플렉서를 더 포함하는 것을 특징으로 하는 부호화 영상데이터의 인터페이스장치.

청구항 15

부호화 영상데이터의 인터페이스장치에 있어서,

어드레스 지정에 따라 데이터의 입/출력이 이루어지는 에스디램;

상기 에스디램의 데이터 리드 및 라이트를 위한 어드레스를 발생하는 어드레스 생성 수단;

부호화 장치로부터 입력된 데이터를 저장하여 상기 에스디램으로 출력하는 입력 인터페싱 수단;

상기 입력 인터페이싱 수단과 입력 및 출력이 이루어지는 입력 버퍼링 수단;

상기 에스디램에 저장된 데이터를 외부채널로 출력하는 출력 인터페이싱 수단;

상기 출력 인터페이싱 수단과 입력 및 출력이 이루어지는 출력 버퍼링 수단;

상기 에스디램의 데이터 입력과 출력을 제어하는 제어신호 생성 수단; 및

현재의 채널버퍼에 저장된 데이터의 정보를 이용하여 가변길이 부호화의 속도를 결정하는 버퍼상태 연산 수단

을 포함하는 부호화 영상데이터의 인터페이스장치.

청구항 16

제 15 항에 있어서,

상기 입력 버퍼링 수단이 에스램(SRAM)으로 이루어진 것을 특징으로 하는 부호화 영상데이터의 인터페이스장치.

청구항 17

제 15 항에 있어서,

상기 출력 버퍼링 수단이 에스램(SRAM)으로 이루어진 것을 특징으로 하는 부호화 영상데이터의 인터페이스장치.

청구항 18

제 15 항에 있어서,

상기 어드레스 생성 수단은,

읽기 어드레스 생성기와 쓰기 어드레스 생성기로 이루어진 것을 특징으로 하는 부호화 영상데이터의 인터페이스장치.

청구항 19

제 15 항에 있어서,

상기 어드레스 생성 수단과 상기 제어신호 생성 수단 사이에 멀티플렉서를 더 포함하는 것을 특징으로 하는 부호화 영상데이터의 인터페이스장치.

청구항 20

부호화 영상데이터의 인터페이스 방법에 있어서,

에스디램을 이용하여 어드레스 지정에 따른 데이터의 입/출력을 수행하는 제 1 단계;

상기 에스디램의 데이터 리드 및 라이트를 위한 어드레스를 발생하는 제 2 단계;

부호화 장치로부터 입력된 데이터를 저장하여 상기 에스디램으로 출력하는 제 3 단계;

상기 에스디램에 저장된 데이터를 외부채널로 출력하는 제 4 단계;

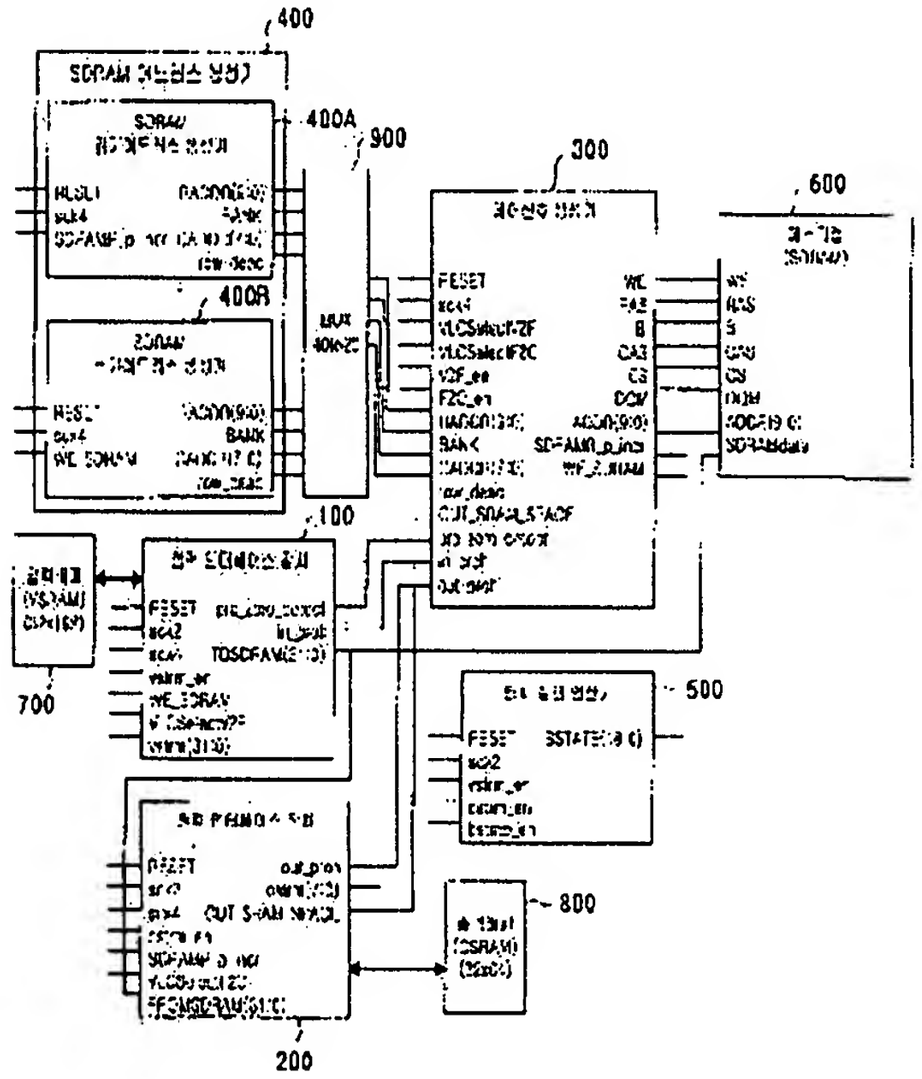
상기 에스디램의 데이터 입력과 출력을 제어하는 제어신호를 생성하는 제 5 단계; 및

현재의 채널버퍼에 저장된 데이터의 정보를 이용하여 가변길이 부호화의 속도를 결정하는 제 6 단계

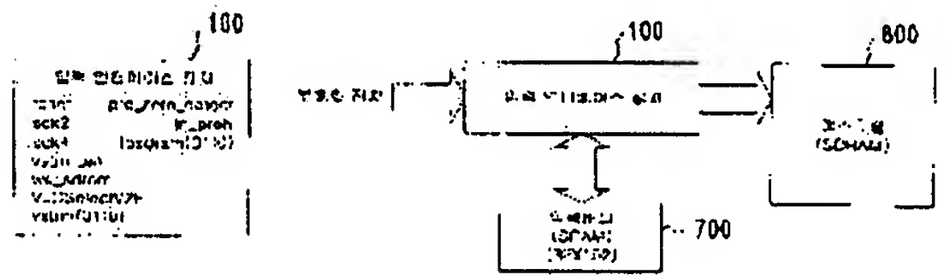
를 포함하는 부호화 영상데이터의 인터페이스 방법.

도면

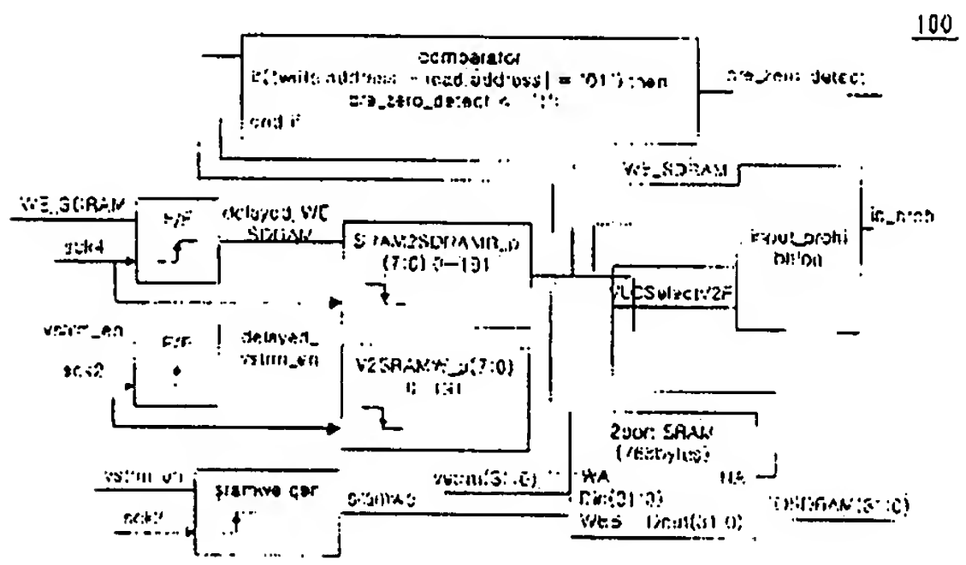
도면1



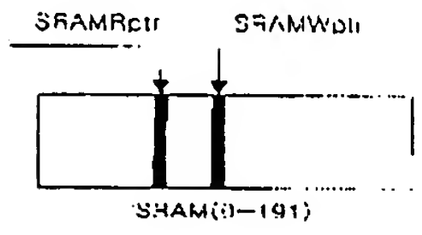
도면2



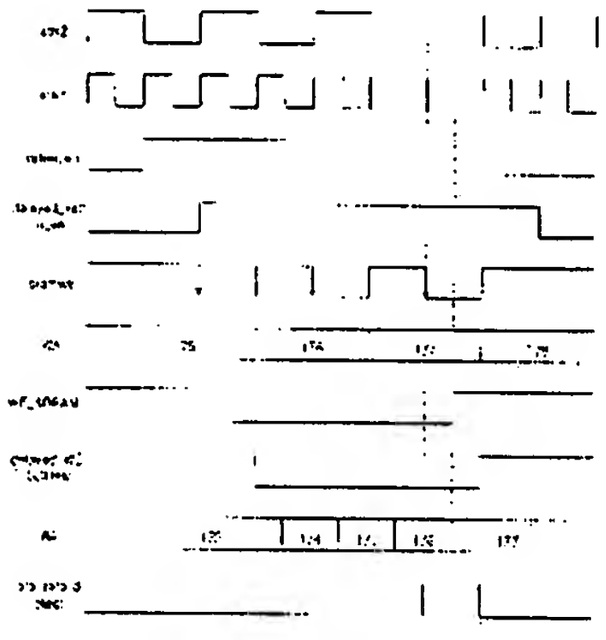
도면3



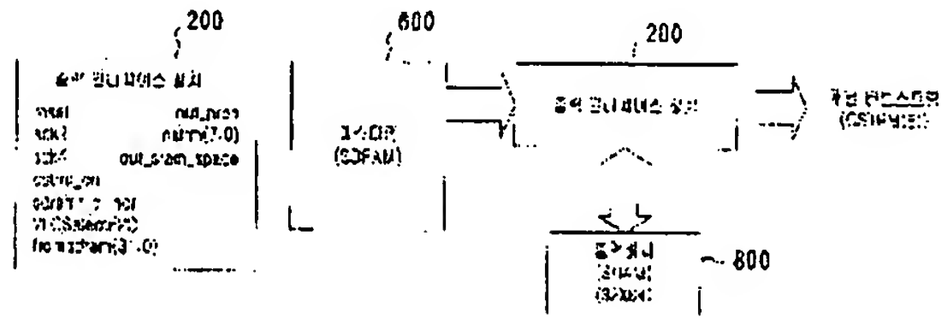
도면4



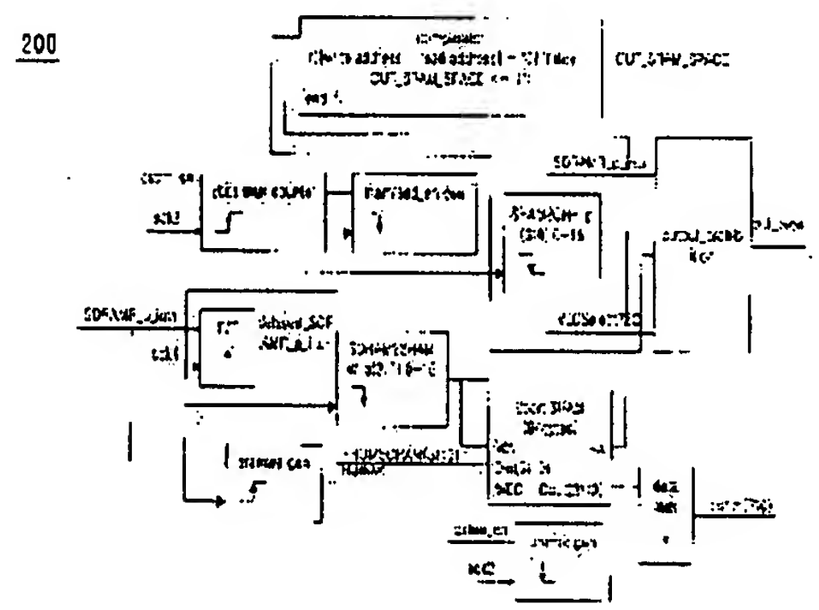
도면5



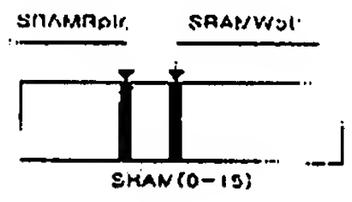
도면6



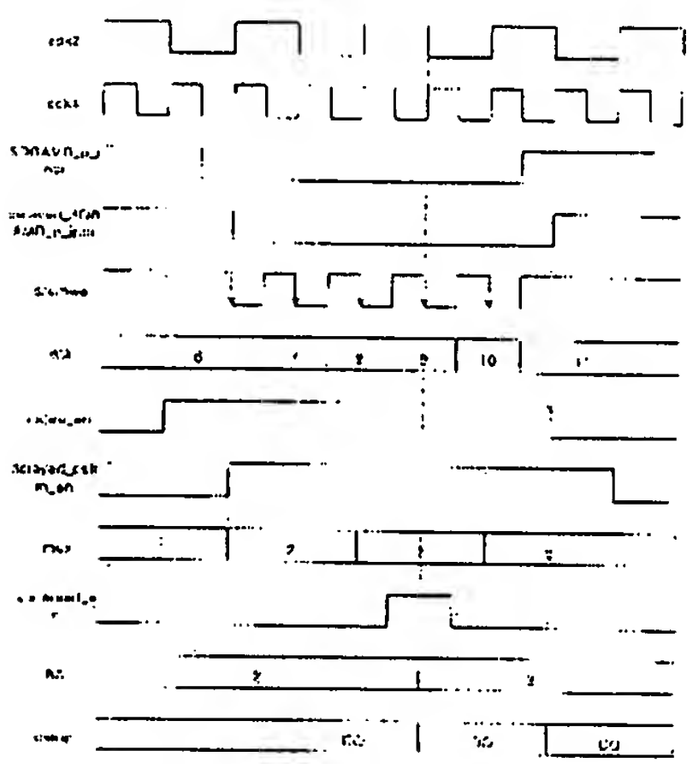
도면7



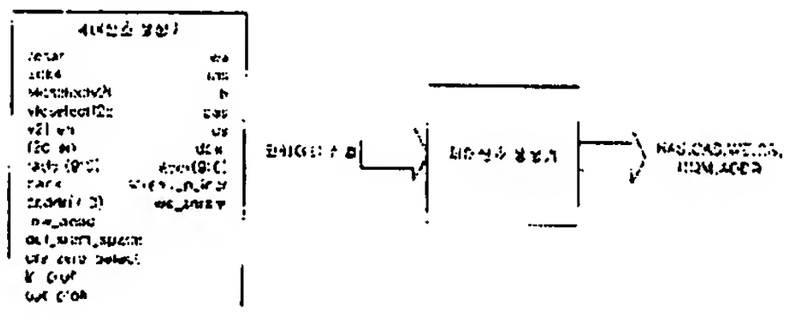
도면8



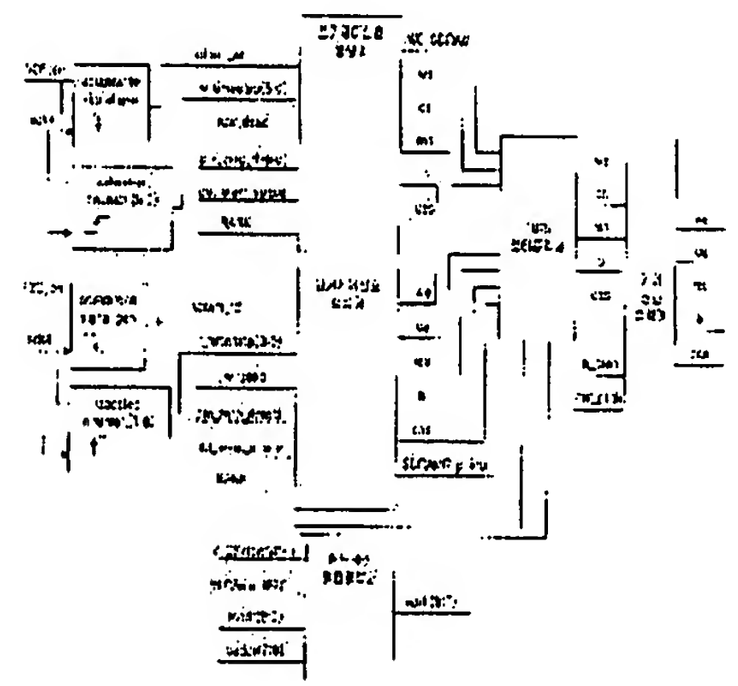
도면9



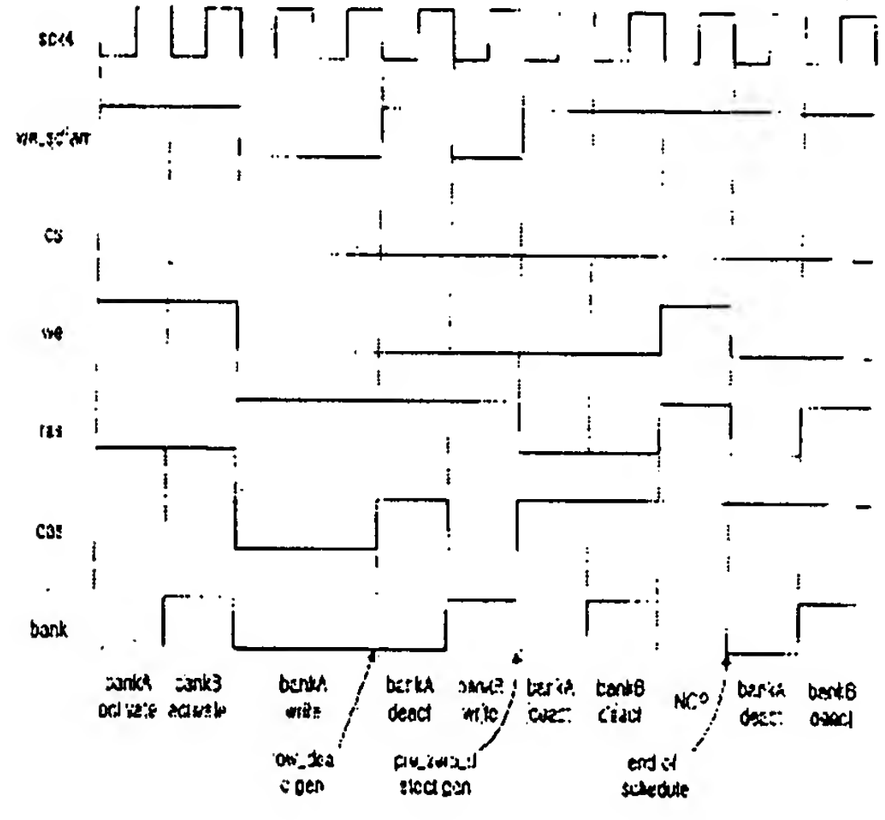
도면10



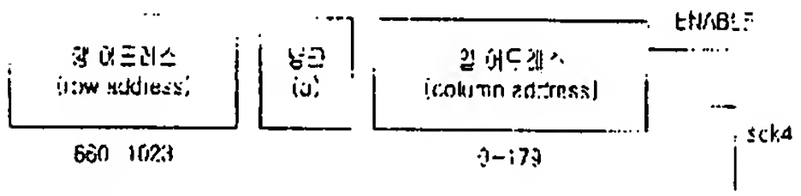
도면11



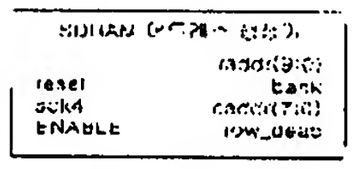
도면12



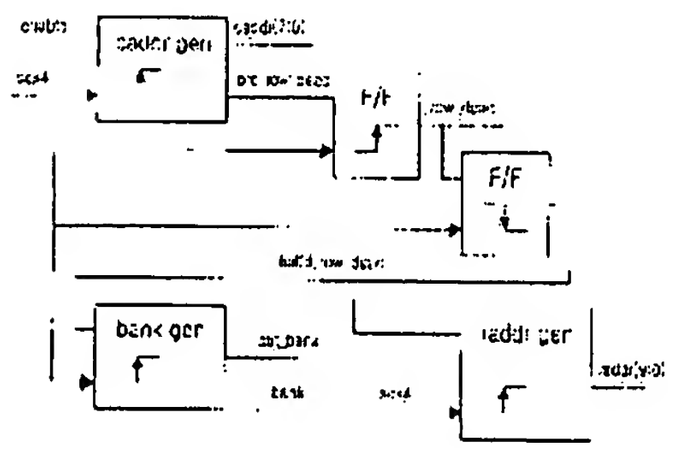
도면13



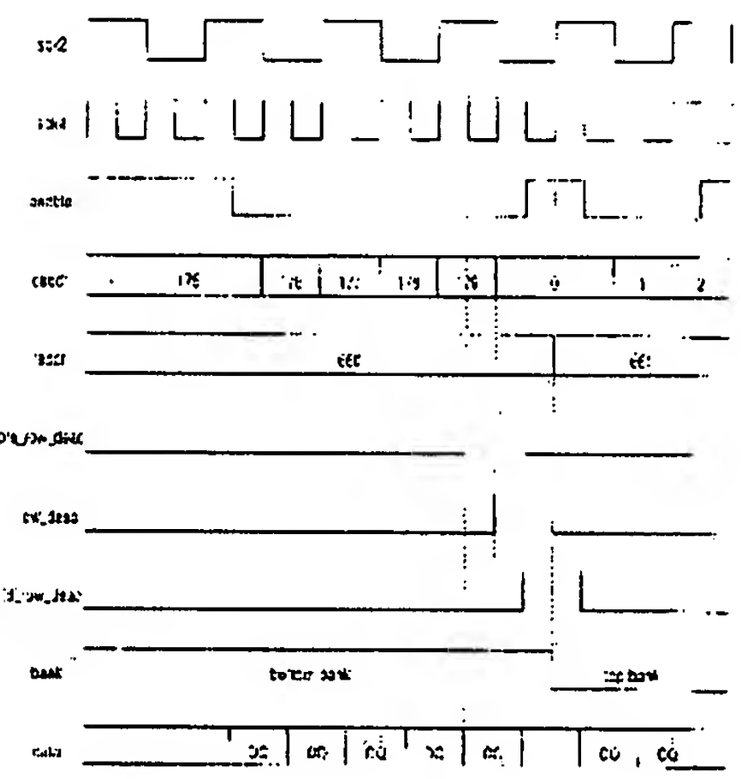
도면14



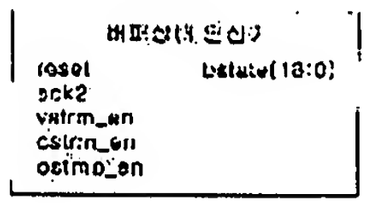
도면15



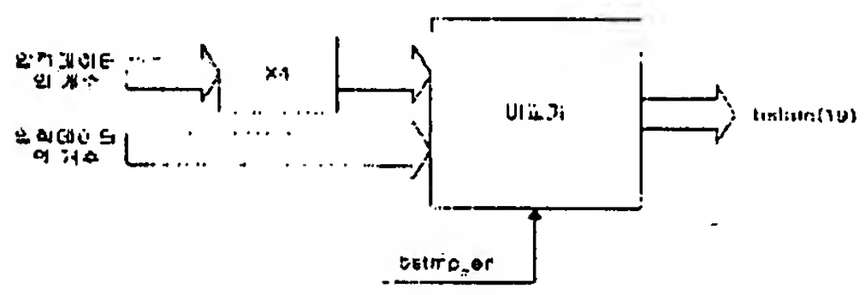
도면16



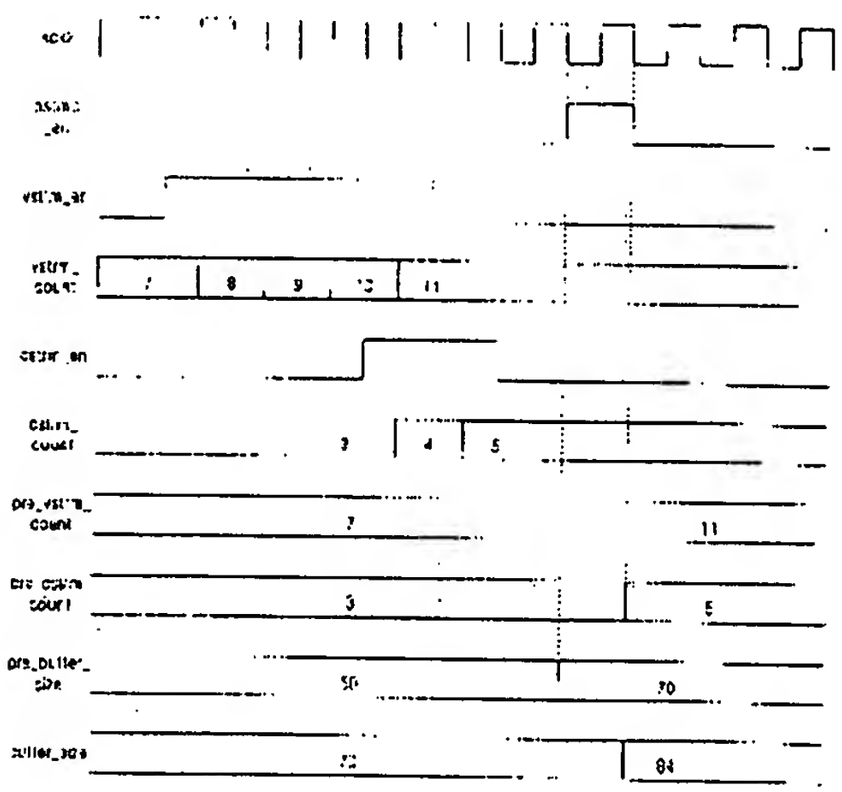
도면17



도면18



도면10


vco
vsync_en
vsync_ar
vsync_count
7
8
9
10
11
osdrn_en
osdrn_count
3
4
5
pre_vsync_count
7
11
pre_osdrn_count
3
5
pre_buffer_size
50
70
buffer_size
72
84